## CLAIMS

What is claimed is:

1. A computer-implemented method of designing an integrated circuit, comprising the steps of:
   a) establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;
   b) establishing a set of software language models for the cores, each software language model implementing an internal algorithm of one of the cores;
   c) establishing a set of hardware description language models for the cores, each hardware description language model implementing an internal logic of one of the cores;
   d) generating software language core interconnection code for inteconnecting the software language models according to the central specification, to generate a software language model of the circuit; and
   e) generating hardware description language core interconnection code for interconnecting the hardware description language models according to the central specification, to generate a hardware description language model of the circuit.

2. The method of claim 1 wherein:
   a) the central specification designates a set of input tokens and a set of output tokens for each core; and
   b) the central specification designates a set of token fields for each interconnection.

Sub A1

3. The method of claim 2 wherein generating the hardware description language core interconnection code comprises generating a set of port declarations from the sets of token fields, input tokens, and output tokens defined in the central specification.

4. The method of claim 2 wherein generating the hardware description language core interconnection code comprises generating an interface port list from the sets of token fields, input tokens, and output tokens defined in the central specification.

5. The method of claim 2 wherein generating the hardware description language core interconnection code comprises generating a set of data type declarations from the sets of token fields, input tokens, and output tokens defined in the central specification.

6. The method of claim 2 wherein generating the hardware description language core interconnection code comprises generating a set of bus definitions from the sets of token fields, input tokens, and output tokens defined in the central specification.

7. The method of claim 2 wherein:
   a) at least one of the token fields defined in the specification includes a token field parameter; and
   b) the central specification further comprises an assignment of a value for the token field parameter.

8. The method of claim 1 further comprising generating a hardware description language declaration for a clock input port and a reset input port for each of the cores.

9. The method of claim 1 wherein, for each core:

a) the central specification defines a set of core parameters; and

b) generating the hardware model comprises generating a set of hardware description language parameter declarations from the set of core parameters.

10. The method of claim 1 wherein:

a) for each core, the central specification defines a set of input tokens and a set of output tokens; and

b) generating the software model comprises generating declarations for a set of pipes and a set of tokens from the sets of input tokens and output tokens defined in the central specification, each pipe serving to transfer tokens between the cores.

11. The method of claim 1 further comprising the step of generating a test bench for the integrated circuit, the test bench including the hardware model and the software model, for comparing a result of a software simulation of the circuit to a result of a hardware simulation of the circuit.

12. The method of claim 11 wherein the test bench comprises a bus driver module for driving the hardware model of the integrated circuit.

13. The method of claim 11 wherein the test bench further comprises a software-driver module for driving the software model of the integrated circuit.

14. The method of claim 11 wherein the test bench further comprises a monitor module for comparing the result of the software simulation to the result of the hardware simulation.

15. The method of claim 11 wherein the test bench further comprises a bus receiver module for requesting tokens from the hardware model.

16. The method of claim 1, further comprising the steps of:

a) driving the hardware model with a simulation hardware input;

b) driving the software model with a simulation software input;

c) detecting a response of the hardware model to the simulation hardware input;

d) detecting a response of the software model to the simulation software input; and

e) comparing the response of the hardware model to the simulation hardware input to the response of the software model to the simulation software input.

17. The method of claim 16 wherein steps (b) and (e) are performed at least in part concurrently with a simulation of the hardware model.

18. The method of claim 1 further comprising the step of generating a synthesis constraint for the circuit.

19. The method of claim 18 wherein the synthesis constraint limits the fraction of a clock cycle used by a core interface.

20. The method of claim 1 wherein generating the hardware description language model comprises the step of instantiating a hardware subcore model forming part of one of the cores.

21. The method of claim 20 wherein generating the software model comprises the step of instantiating a software subcore model forming part of one of the cores.

22. The method of claim 1 wherein generating the software model comprises the step of instantiating a software subcore model forming part of one of the cores.

23. The method of claim 1 wherein the plurality of interconnections comprises a dedicated interconnection connecting a first core and a second core, the first interconnection comprising:

a) a ready connection for carrying a ready signal from the first core to the second core, the ready signal being indicative of a readiness of the first core to transmit a token to the second core;

b) a request connection for carrying a request signal from the second core to the first core, the request signal being indicative of a readiness of the second core to receive the token from the first core; and

c) a token bus for transmitting the token from the first core to the second core upon a synchronous assertion of the ready signal and the request signal.

24 A computer-implemented method of designing an integrated circuit, comprising the steps of:

a) establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;

b) for each core, establishing hardware description language template code for an internal logic of the core;

c) for each core, establishing software language template code for an internal algorithmic functionality of the core;

d) processing the central specification to generate software language core interconnection code for interconnecting the software language template code for the cores, to generate a software language model of the circuit; and

e) processing the central specification to generate hardware description language core interconnection code for interconnecting the hardware description language template code for the cores, to generate a hardware description language model of the circuit.

25. The method of claim 24 wherein the hardware description language core interconnection code comprises an interface port list, a set of port declarations, a set of interface bus definitions, and a set of data type declarations.

Sub a1

26. The method of claim 25 wherein the port list comprises a global clock port and a reset port for each core.

27. The method of claim 24 wherein the software language core interconnection code comprises a set of pipe declarations and a set of token declarations.

28. The method of claim 24, further comprising the step of processing the central specification to add a parameter list to the hardware description language template code.

29. The method of claim 24, further comprising the step of processing the central specification to add a parameter list to the software language template code.

30. A computer-implemented method of designing an integrated circuit, comprising the steps of:

a) establishing a central specification for the integrated circuit, the central specification designating a plurality of cores and a plurality of dedicated interconnections between the cores;

b) generating a software language, functional model of the integrated circuit from the central specification and from a set of software language, functional models of the cores;

c) generating a hardware description language model of the integrated circuit from the central specification and from a set of hardware description language models of the cores;

d) generating a test bench for the circuit from the hardware description language model and the software

Sub a1

language model, for comparing a result of a software simulation of the circuit to a result of a hardware simulation of the circuit; and

e) generating a set of logic synthesis constraints for the circuit from the central specification, for constraining a logic synthesis of the circuit.

31. A computer-implemented method of designing an integrated circuit, comprising the steps of:

a) establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;

b) generating a software language, functional model of the integrated circuit from the central specification and from a set of software language, functional models of the cores; and

c) generating a hardware description language model of the integrated circuit from the central specification and from a set of hardware description language models of the cores.

32. A computer system programmed to perform the steps of:

a) establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;

b) generating a software language, functional model of the integrated circuit from the central specification and from a set of software language, functional models of the cores; and

c) generating a hardware description language model of the integrated circuit from the central

~~specification and from a set of hardware description language models of the cores.~~

~~33. A computer-readable medium encoding instructions to perform the steps of:~~

~~a) establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;~~

~~b) generating a software language, functional model of the integrated circuit from the central specification and from a set of software language, functional models of the cores; and~~

~~c) generating a hardware description language model of the integrated circuit from the central specification and from a set of hardware description language models of the cores.~~

~~34. A computer-readable medium encoding a representation of an integrated circuit designed by the method of claim 31.~~

35. An integrated circuit designed by ~~the~~ method of claim 31.

~~36. A circuit design apparatus comprising:~~

~~a) a storage device for establishing a central specification for the integrated circuit, the central specification designating a plurality of data-driven cores and a plurality of interconnections between the cores;~~

~~b) software interconnection means for generating a software language, functional model of the integrated circuit from the central specification and from a set of software language, functional models of the cores; and~~

Sub a1

c) ~~hardware~~ description language interconnection means for generating a ~~hardware~~ description language model of the integrated circuit from the central specification and from a set of hardware description language models of the cores.

add a1